by the processor pipeline, the indicator elements being architecturally addressable when the processor pipeline is executing under one of the architectures or data storage conventions, and architecturally unaddressable when the processor pipeline is executing under the other architecture or data storage convention;

the memory unit and/or processor pipeline further designed to recognize an execution flow from the first page, whose associated indicator element indicates the first architecture or execution convention, to the second page, whose associated indicator element indicates the first architecture or execution convention, and in response to the recognizing, to adapt a processing mode of the processor pipeline or a storage content of the memory to effect execution of instructions in the architecture and/or under the convention indicated by the indicator element corresponding to the instruction's page.

27. (amended) The method of claim 22, wherein

a rule for copying data from the first location to the second is determined by examining a descriptor associated with the instruction before the recognized execution flow or transfer.

59. (amended) The method of claim 54, wherein

a rule for copying data from the first location to the second is determined by examining a descriptor associated with the instruction before the recognized execution flow or transfer.

93. (amended) The method of claim 87, wherein

a rule for copying data from the source memory or register to the destination register or memory is determined by examining a descriptor associated with the address of the control-transfer instruction.

94. (amended) A method, comprising the steps of:

executing a section of computer object code twice, without modification of the code section between the two executions, the code section materializing a destination address into a register and being architecturally defined to directly transfer control indirectly through the register to the destination address, the two executions materializing two different destination addresses;

the two destination code sections at the two materialized destination addresses being coded in two distinct instruction sets and, respectively, obeying the default data storage conventions native to each of the two instruction sets, neither instruction set being a subset of the others.

97. (amended) The computer processor of claim 96, wherein the memory unit and software are designed to effect a transition between instruction boundaries, between execution in a region coded in the first instruction set using the first data storage convention to execution in a region coded in the second instruction set using the second data storage convention, so that code at the source of the flow or transfer may effect the execution transition without being specially coded for code at the destination.

98. (amended) The microprocessor chip of claim 96, wherein the two data storage conventions are first and second calling conventions.

100. (amended) The microprocessor chip of claim 98, further comprising:

software and/or hardware designed to copy a datum from a first location to a second location, the first location having a use under the first calling conventions analogous to the use of the second location under the second calling conventions.

102. (amended) The computer processor of claim 98, wherein

a rule for altering the data storage content from the first calling convention to the second calling convention is determined based on an instruction at the location of execution at the source of the recognized execution flow or transfer.

103. (amended) The computer processor of claim 98, wherein

a rule for altering the data storage content from the first calling convention to the second calling convention is determined by examining a descriptor associated with the instruction before the recognized execution flow or transfer.